Communicating P Systems: Bio-inspired Computational Models for Complex Systems

Erzsébet Csuhaj-Varjú

Department of Algorithms and Their Applications, Faculty of Informatics,
Eötvös Loránd University, Budapest, Hungary
csuhaj@inf.elte.hu

Abstract: The theory of P systems or membrane systems, a vivid scientific area in bio-inspired computing, deals with computational models inspired by architecture and functioning of living cells and tissues, and neural systems. This paper contains ideas on why and how purely communicating P systems (important variants of P systems) can be interpreted as complex natural systems. We give a summary of the most relevant results concerning these P systems and provide their interpretation in terms of complex systems. We propose open problems and new directions for future research.

1 Introduction

P systems or membrane systems are distributed parallel computing devices, inspired by architecture and functioning of living cells. The original concept of a P system was introduced by Gheorghe Păun in 1998, for details see the seminal paper [17]. Later the concept has been extended to model living tissues and neural systems. Recently, membrane computing and its applications are a vivid, well-established scientific area in natural computing and related fields of computer science.

The basic model, the cell-like system consists of a hierarchically embedded structure of membranes where each membrane encloses a region which contains objects and may also contain other membranes (regions). The regions are associated with rule sets. These rules describe the evolution and communication of the objects present in the regions. P systems can also be considered as distributed parallel multiset rewriting systems, since the rules describe the evolution and/or communication of multisets of objects in the actions. The P system operates with changing its configurations, in other words, with transitions from one configuration to some other one. A configuration of a P system is the underlying graph structure of its regions and the multisets of objects in the regions (in other words, compartments, nodes of the graph). The architecture of a cell-like P systems, called also a transition P system or symbol-object P system, has a tree as underlying graph, has a rooted tree architecture. In case of tissue P systems, and P systems modeling neural systems the underlying graph is an arbitrary graph.

A computation is a sequence of configurations where the subsequent elements are obtained by transitions and the sequence starts with a so-called initial configuration. The result of the computation is usually defined by the number of objects found in the so-called output region when the operation of the P system halts (no rule in any of the regions can be performed). Instead of sets of numbers, sets of vectors of multiplicities of objects can also be considered as result.

P systems have intensively been studied during the years; for concepts, results, and a detailed overview of the area consult [18].

Since one of the main goals of introducing the notion of a P system was to define a bio-inspired distributed computing device that is as powerful as Turing machines and combines properties of natural systems and classical models of (distributed) computation, the investigations in the field have mainly focused on computability and complexity questions. Recently, other areas of the theory of computation, algorithms, simulations, applications in computer science have enhanced the research scope, only mentioning a few. Since P systems exhibit properties of both qualitative and quantitative models, features, approaches and aspects of this scientific area can be compared to that of various disciplines.

Studying how P systems can be used for modeling natural, complex systems is a reasonable research direction from this point of view. Some initial ideas on connections between natural complex systems and P automata (particular variants of P systems) can be found in [4]. In the following we discuss the relation between purely communicating P systems (important variants of P systems) and complex systems, with the approach used in [4] and continue developing the ideas presented in that paper.

We first provide a brief description of interpretation of components and properties of cell-like P systems in terms of complex systems. Then we present such a comparison in case of symport/antiport P systems and generalized communicating P systems, both models are purely communicating variants of membrane systems. We close the paper with some conclusions.

2 Complex Systems and P Systems

Complex systems theory is an important scientific area studying how parts of a system give rise to the collective

behaviors of the system, and how the system interacts with its environment. One of the main questions in the area is whether or not the system as a whole is more powerful than the sum of the power of their individual components. Examples for complex systems are, for example, social systems, social networks formed out of people, molecules formed out of bio-chemical ingredients, systems of agents, only to mention a few. Thus, the study of complex systems is an interdisciplinary scientific field which focuses on questions about wholes, parts, and relationships. For detailed information the reader is referred to [2], [15].

Examining cell-like P systems, we can conclude that the above properties and features of complex systems are characteristic of P systems as well. Objects of a P system can be considered as elementary agents, elementary ingredients of the complex system; multisets of objects form communities of agents or collections of ingredients. Regions (or compartments) of the P system correspond to components of the complex system, however not elementary components. The rules associated to the regions represent interactions between communities of agents, they can be classified according to their forms and application mode. The rules describe local activities, since they affect either the region in which they are executed or the neighboring regions. The neighboring regions of the P system represent components of the complex system being in close connection (neighboring components), and the environment itself can also be considered as a special, distinguished component. The P system operates with applying its rules in the same way as the complex system operates with interactions between its components and its environment. The behavior of a P system can be described by its configuration (state) sequences or the result of a halting configuration sequence which starts from an initial configuration. Analogous description can be used in case of complex systems as well. This approach provides the option to identify behavioral patterns and related properties as well. By behavioral complexity we mean the complexity of the description. For example, a family of P systems computing the recursively enumerable family of numbers is a family of complex systems that exhibits maximal complexity.

Various variants of P systems are as powerful as Turing machines, thus provide maximal computational power, and in terms of complex systems exhibit behavior of maximal complexity. An important problem area is whether or not restricted P systems are able to exhibit such complex behavior. Furthermore, how we can interpret them in terms of complex systems. In the following, we deal with these questions.

Throughout the paper we assume that the reader is familiar with the theory of computation, thus we recall only a very few notations and notions. An alphabet V is a finite nonempty set; the set of all strings over V is denoted by V^*, and λ denotes the empty word. A finite multiset over an alphabet V is a mapping $M : V \to \mathbf{N}$ where $\mathbf{N}$ is the notation for the set of non-negative integers; $M(a)$ is said to be the multiplicity of a in M. A finite multiset M can also be represented by any string $x \in V^*$ where $|x|_a = M(a)$ for all $a \in V$ (clearly, for a given string x, any permutation of x represents the same multiset).

3 Symport/antiport P Systems

In the case of transition P systems (the basic, cell-like model), the multisets of objects are able to change (to be rewritten) and to move across the membranes, i.e., the agents can evolve and can be communicated. In the following we will discuss purely communicating P systems, where objects can only be transported between neighboring compartments (including the environment), i.e., only communication of agent communities can be performed.

P systems having rules that only allow to move multisets of objects from one compartment to one of its neighbors or simultaneously in opposite directions across the membrane, are called symport P systems or antiport P systems, respectively; we use term symport/antiport P systems for these two types of P systems. These notions have biological motivation and were introduced in [16].

Formally, a P system with symport and/or antiport rules (a symport/antiport P system, for short) is a construct $\Pi = (O, T, E, \mu, (w_1, R_1), \ldots, (w_n, R_n,), i_0)$, where $n \geq 1$, O is the finite alphabet of objects; $T \subseteq O$ is the alphabet of terminal objects; $E \subseteq O$ is the set of objects in the environment which appear in infinitely countable number of copies; μ is a membrane structure of n membranes, where 1 indicates the skin membrane; $w_i \in V^*$, $1 \leq i \leq n$, is the initial contents (finite multiset) of region i; R_i is a finite set of rules associated to membrane i, $1 \leq i \leq n$. The rules are of one of the forms $(u, out; v, in)$ with $u \neq \lambda$, $v \neq \lambda$ (antiport rule), (u, in), (u, out) with $u \neq \lambda$ (symport rules,) where $u, v \in O^*$; $i_0 \in \{1, \ldots, n\}$ is the label of an elementary membrane, called the output membrane.

The skin membrane separates the P system from the environment; its region is in the root of the tree representing the architecture, the structure of the system.

An antiport rule, $(u, out; v, in) \in R_i$, $1 \leq i \leq n$, exchanges multiset of objects u in region i with multiset v from the parent region of i. The symport rule (u, out) moves multiset u out of region i, to its parent region, and symport rule (u, in) imports u from the parent region into region i. If the rules are associated with the skin membrane, then the parent region is the environment. In the case of symport rules, if the parent region of i is the environment, then the imported multiset must contain at least one symbol not in E (otherwise an infinite number of objects would enter in the system). Multisets w_i, $1 \leq i \leq n$, and the membrane structure μ represent the initial configuration of Π. At the beginning, the environment does not contain any element of $O \setminus E$.

Symport/antiport rules can also be associated with promoters and/or inhibitors. In this case the objects are allowed to be transported between the regions, if the corresponding regions have (do not have) the promoter (in-

hibitor) multiset included in their current multiset of objects.

Similarly to other types of P systems, symport/antiport P systems compute by performing transitions between subsequent configurations. The configurations consists of the multisets of objects that belong to the regions and the multiset of the non-environmental objects (not elements of E) which appear in the environment. Symport/antiport P systems usually use the non-deterministic maximally parallel working mode, but other derivation modes have also been considered (for more details, see [18, 12]). A sequence of transitions starting with the initial configuration is a computation.

The result of the computation in Π is the number of objects from T that can be found in region i_0 when the system halts. If no terminal alphabet is distinguished, then the number of all objects in i_0 at the end of a halting computation is the result of the computation.

Instead of sets of numbers, sets of vectors of multiplicities of elements of T can also be considered as results. Symport/antiport P systems can also be not only generating, but accepting devices. In this case, the input is the initial contents of a distinguished region. The symport/antiport P system accepts the input if starting with this initial configuration it enters a halting configuration by a computation.

It can be observed that symport/antiport P systems can be considered as models for complex systems. The objects in the membrane architecture represent very simple, elementary components of the system, the agents, collections of which interact with each other and with the environment of the system. The interactions, defined by the symport/antiport rules, are local interactions. Locality arises from the fact that both symport and antiport rules involve only neighboring regions (we consider the environment as the parent region of the skin region). Antiport P systems behave in non-linear manner, since they behave in different ways to the same input multisets from the environment depending on their current state.

Symport/antiport P systems are computationally complete computing devices [16]. Furthermore, for any register machine, a P system of this type with only one region (one membrane) can be constructed such that it simulates the register machine. Extensive investigations proved that symport/antiport P systems with small size parameters are as powerful as register machines, i.e., are computationally complete. These parameters are, for example, the number of regions, the number of rules in the system, the number of objects in the multisets of the rules (for more details on these results, see [18]).

Notice that the fact that symport/antiport P systems are computationally complete prove that these systems as formal models of complex systems demonstrate emergent behaviour since the power of the system as a whole significantly exceeds the power of a single component without any interaction. Furthermore, like in case of Turing machines, universal antiport P systems can be constructed. Moreover, their size parameters are bounded by constants [6]. Universal antiport P systems correspond to certain "core" complex systems.

Recall that the result of the computation of symport/antiport P systems is the number of (terminal) objects or the set of vectors of (terminal) objects in the output region in a halting configuration. However, the computation can also be described by the sequence of multisets which enter the P system during its functioning. This observation inspired to develop the concept of a P automaton [7], a variant of accepting purely communicating P systems. In this case, the input sequences of multisets imported by the symport/antiport P system from the environment are mapped onto words over a finite alphabet, thus, form a language. The mapping is “easily” computable, usually computable in linear space. A similar notion, called an analysing P system, [13] uses another mapping to define words of the language, the mapping that orders to every multiset the set of words which are permutations of all elements of the multiset. This mapping is denoted by f_{perm}. The concept of a P automaton combines features of a P system and that of variants of classical automata. Several intersting results have been obtained on this model. For example, P automata working in the non-deterministic maximally parallel manner and with mapping computable in linear space describe the class of context-sensitive languages, while using mapping f_{perm}, it defines a class of languages of sublogarithmic space complexity. For more information the reader is referred to [18, 5].

P automata are models of complex natural systems (a detailed discussion on the topic can be found in [4]). To prove that the statement holds, we recall some observations from [4]).

An important property of complex systems is that the interactions between the agents and the environment may imply changes in the system itself. This is exactly the case for P automata, since the multisets of objects entering the system from the environment can significantly change the coming configuration sequence. The P automaton as complex system is an open system, since (multisets) of agents, i.e., (multisets of) objects can join and leave the system via symport/antiport rules. The standard P automaton is given with a static membrane structure, that is, its architecture does not change during the functioning of the system, which is a rather restrictive condition. Examples for P automata with dynamically changing membrane structure are the P automata with marked membranes [8]) (inspired by biology) and the active P automata [3] (motivated by natural language processing), however in these cases the underlying P systems is not a symport/antiport P system.

P automata can also be considered as tools for describing languages over infinite alphabets without any extension or additional component added to the construct, since in the case of maximally parallel rule application the number of objects entering the skin region not necessarily can be bounded by a constant. This means that in this case the agent population can increase fast and the number of

simultaneous interactions with the environment cannot be bounded by any constant.

It would be worth studying P automata with adding the features of membrane creation, dissolution, division [18], P automata with dynamically varying structure. The obtained new variants can be used for modeling self-configurating systems and systems re-configurating theirselves under control coming from outside, since both the objects inside the regions and the objects entering the system from the environment can launch a re-configuration in the membrane structure.

4 Generalized Communicating P Systems

As we discussed in the previous section, purely communicating P systems like symport/antiport P systems are of particular interest in membrane computing, since several variants of P systems with only communication rules are computationally complete, thus these restricted systems are able to obtain maximally complex behavior (with respect to descriptions by computable sets of numbers, sets of vectors of natural numbers, and languages). An intriguing question is what can we say about the type and form of interactions between the agents in these systems, how much extent the interactions can be simplified.

Generalized communicating P systems give an answer to this question. We note that the concept was originally introduced with the aim of providing a common generalization of various purely communicating models [19].

A generalized communicating P system (a GCPS for short), is a tissue-like P system where each node represents a cell and each edge is represented by a rule (an interaction). Each node contains a multiset of objects that can be communicated, i.e., it may move between the cells according to interaction (communication) rules.

The form of an interaction rule is $(a,i)(b,j) \to (a,k)(b,l)$ where a and b are objects and i,j,k,l are labels identifying the input and the output cells. Such a rule means that an object a from cell i and an object b from cell j move synchronously (in one step) to cell k and cell l, respectively. These interactions are particularly simple, since there are only two objects (two agents) involved in them.

Formally, a generalized communicating P system (a GCPS) of degree n, where $n \geq 1$, is an $(n+4)$-tuple $\Pi = (O,E,w_1,\ldots,w_n,R,h)$ where O is an alphabet, called the set of objects of Π; $E \subseteq O$; called the set of environmental objects of Π; $w_i \in O^*$, $1 \leq i \leq n$, is the multiset of objects initially associated to cell i; R is a finite set of interaction rules or communication rules of the form $(a,i)(b,j) \to (a,k)(b,l)$, where $a,b \in O$, $0 \leq i,j,k,l \leq n$, and if $i = 0$ and $j = 0$, then $\{a,b\} \cap (O \setminus E) \neq \emptyset$; i.e., $a \notin E$ and/or $b \notin E$; and $h \in \{1,\ldots,n\}$ is the output cell.

The generalized communicating P system is embedded in an environment, called cell 0, which may have certain objects in an infinite number of copies and certain objects only in a finite number of copies. The GCPS and the environment interact by using the rules given above, with the restriction that at every computation step only a finite number of objects are communicated to each cell by the environment. As usual in P systems' theory, the rules are applied in a maximally parallel manner. This implies a possible change in the current state of the GCPS (the multisets representing the contents of the cells). A computation in a GCPS is a sequence of states directly following each other that starts from the initial state and ends in a halting state. The result of the computation is the number of objects found in a distinguished cell, the output cell.

It can be seen that GCPSs are formal models of complex systems as well: the objects correspond the elementary components, agents and the system operates with the interactions of agents. However, in this case the nodes are not explicitly given, instead they are defined by the interaction rules between the agents. In this way, the architecture of the system emerges in the course of functioning.

To apply an interaction rule only two objects and at most four locations (nodes) are involved, thus it is worth studying the form of interaction rules.

We recall the possible restrictions on the interaction rules (modulo symmetry). The following cases are distinguished, and then the systems is called GCPSs with minimal interaction (for details see [10]. The reader may easily observe that these rules represent minimal interaction patterns.

1. $i = j = k \neq l$: the conditional-uniport-out rule (the *uout* rule) sends b to cell l provided that a and b are in cell i;

2. $i = k = l \neq j$: the conditional-uniport-in rule (the *uin* rule) brings b to cell i provided that a is in that cell;

3. $i = j$, $k = l$, $i \neq k$: the symport2 rule (the *sym*2 rule);

4. $i = l$, $j = k$, $i \neq j$: the antiport1 rule (the *anti*1 rule) , i.e., a and b are exchanged in cells i and k;

5. $i = k$ and $i \neq j$, $i \neq l$, $j \neq l$: the presence-move rule (the *presence* rule) moves the object b from cell j to l, provided that there is an object a in cell i and i,j,l are pairwise different cells;

6. $i = j$, $i \neq k$, $i \neq l$, $k \neq l$: the *split* rule sends a and b from cell i to cells k and l, respectively;

7. $k = l$, $i \neq j$, $k \neq i$, $k \neq j$: the *join* rule brings a and b together to cell k;

8. $l = i$, $i \neq j$, $i \neq k$ and $j \neq k$: the *chain* rule moves a from cell i to cell k while b is moved from cell j to cell i, i.e., to the cell where a located previously;

9. i,j,k,l are pairwise different numbers: the parallel-shift rule (the *shift* rule) moves a and b from two different cells to another two different cells.

During the years, GCPSs have been studied in detail. The investigations have mainly been focused on their computational power and their relation to other models like Petri nets.

It has been shown that GCPSs in general, and even with minimal interaction are able to generate any recursively enumerable set of numbers. Furthermore, to obtain computational completeness a relatively small number of cells and a simple underlying hypergraph architecture is sufficient [10, 14, 11]. Thus, for example GCPSs with three cells and with only join, or only split, or only chain rules are computationally complete computing devices [11]. It is also shown that the maximal expressive power can also be obtained with GCPSs where the alphabet of objects is a singleton [9]. Moreover, computational completeness with small number of cells can also be obtained if the objects of the environment are provided with a rewriting system generating multisets of objects step by step [1].

If we consider GCPSs as formal models of complex systems, we may derive interesting consequences. Namely, the maximal complexity of the collective behavior of the system (represented by the generated sets of numbers) can be obtained with very simple, uniform interaction patterns, with only one type of elementary actions. Furthermore, at any moment of operation the number of groups, collections of agents is a very small number. It is also proved that in case of certain type of the interaction rules there is no need to distinguish different types of agents in order to obtain the collective behavior with maximal complexity. Furthermore, these systems are able to tolerate the changes in the environment not caused by actions of the agents.

Although several questions and problems concerning generalized communicating P systems have been investigated, some ideas would be worth studying. For example, what can we say about GCPSs where the interaction rules change in time. Namely, the new locations of the objects depend on the number of performed computation steps. One other interesting aspect could be to introduce concepts from evolutionary computing in this area: those interactions which are not used or rarely used change their form or are dismissed from the set of rules. It also would be interesting to examine the concept of similarity in case of these systems, especially concerning behavior or the sequence of (multi)sets of performed interactions.

5 Conclusion and Open Problems

P systems, even purely communicating variants, can be considered as formal models of natural complex systems, since they are dynamically changing systems which are in communication (interaction) with their environments. To explore this relation, further investigations are needed to interpret such concepts as emergent phenomena, nonlinearity, interaction complexity, behavioral complexity in P systems theory, in case of different variants of P systems. Especially interesting problems are to define these notions for P systems with dynamically changing underlying architecture, with division and membrane creation. The relation between P systems with infinite runs and complex systems would also be a challenging topic for future research.

6 Acknowledgement

This work was supported by NKFIH (National Research, Development, and Innovation Office, Hungary) Grant no. K 120558.

References

[1] Balaskó, Á., Csuhaj-Varjú, E., Vaszil, G.: Dynamically Changing Environment for Generalized Communicating P Systems. In: Rozenberg, G. et. al (eds.) Membrane Computing - 16th International Conference, CMC 2015, Valencia, Spain, August 17-21, 2015, Revised Selected Papers. LNCS 9504, Springer, 2015, pp. 92–105

[2] Boccara, N.: Modeling Complex Systems. Graduate Texts in Physics, Springer (2010)

[3] Bel-Enguix, G., Gramatovici, R.: Parikh mapping and iteration. In: Ciobanu, G. et al (eds.): Applications of Membrane Computing. Natural Computing Series 2235, Springer, Berlin, 2006, 389—410

[4] Csuhaj-Varjú, E.: P Automata: Automata-like constructs modeling complex natural systems. Bensch, S. et al (eds.): Fifth Workshop on Non-Classical Models for Automata and Applications - NCMA 2013, Umea, Sweden, August 13 - August 14, 2013, Proceedings. books@ocg.at 294, Österreichische Computer Gesellschaft 2013, ISBN 978-3-85403-294-6, 2013, 13–30

[5] Csuhaj-Varjú, E.: P and DP Automata: Unconventional versus Classical Automata. Int. J. Found. Comput. Sci. **24(7)** (2013) 995–1008

[6] Csuhaj-Varj, E., Margenstern, M., Vaszil, G., Verlan, S.: On small universal antiport P systems. Theor. Comput. Sci. **372(2-3)** (2007) 152–164

[7] Csuhaj-Varjú, E., Vaszil, G.: P Automata or Purely Communicating Accepting P Systems. In: Păun G. et al (eds): Membrane Computing. WMC 2002. LNCS, vol 2597. Springer, Berlin, Heidelberg, 2003, 219–233

[8] Csuhaj-Varjú, E., Vaszil, G.: (Mem)brane automata. Theor. Comput. Sci. **404 (1-2)** (2008) 52–60.

[9] Csuhaj-Varjú, E., Vaszil, G., Verlan, S.: On generalized communicating P systems with one symbol. In: Gheorghe, M. et al (eds.) 11th International Conference on Membrane Computing, 2010, Jena, Germany, LNCS 6501, Springer, 2010, pp. 160–174.

[10] Csuhaj-Varjú, E., Verlan, S.: On generalized communicating P systems with minimal interaction rules. Theor. Comput. Sci. **412** (2011) 124–135

[11] Csuhaj-Varjú, E., Verlan, S.: Computationally Complete Generalized Communicating P Systems with Three Cells. In: Gheorghe, M. et al (eds.): Membrane Computing. CMC 2017. LNCS 10725. Springer, Cham., (2018), 118–128

[12] Freund, R.: How derivation modes and halting conditions may influence the computational power of P systems. J. Membr. Comput. **2 (1)** (2020), 14–25.

[13] Freund, R., Oswald, M.: A short note on analysing P systems. Bull. EATCS **78** (2002) 231–236

[14] Krishna, S. N., Gheorghe, M., Ipate, F., Csuhaj-Varjú, E., Ceterchi, R.: Further results on generalised communicating P systems. Theor. Comput. Sci. **701** (2017) 146–160

[15] Northrop, R. B.: Introduction to Complexity and Complex Systems. CRC Press, Taylor and Francis Group, Roca Baton (2010)

[16] Păun, A., Păun, G.: The Power of Communication: P Systems with Symport/Antiport. New Gener. Comput. **20(3)** (2002) 295–306

[17] Păun, G.: Computing with Membranes. J. Comput. Syst. Sci. **61(1)** (2000) 108–143

[18] Păun, G., Rozenberg, G., Salomaa, A. (eds.): The Oxford Handbook of Membrane Computing. Oxford University Press (2010)

[19] Verlan, S., Bernardini, F., Gheorghe, M., Margenstern, M.: Generalized communicating P systems. Theor. Comput. Sci. **404(1-2)** (2008) 170–184